Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC). Vancouver, Canada, October 22-25, 1995.

MODES IN HUMAN-AUTOMATION INTERACTION: INITIAL OBSERVATIONS ABOUT A MODELING APPROACH

Asaf Degani
Georgia Institute of Technology
Atlanta, GA
and San Jose State University
San Jose, CA

Alex Kirlik
Georgia Institute of Technology
Atlanta, GA

ABSTRACT

This paper is an exploration into the relationship between the operational environment, the human supervisor, and the mode structure of modern control systems. Based on a field study describing operators' (e.g., pilots, controllers, technicians) interaction with modal systems, we developed several hypotheses about why and how operators transition among modes. We used these hypotheses to develop a framework of the complete environment-human-machine relationship. The framework, called "OFAN," is based on Statecharts and Operator-Function models—both modern extensions to the finite-state-machine theory.

Using the OFAN framework, we describe two examples of moding problems and identify the particular features that induce such problems. In the first example, a moding problem in a display, it was the product of dual transitions into a state: one consistent with the layout of the controls (and therefore intuitive), the other dependent upon some internal state (and therefore unintuitive). In the second example, a moding problem in an automatic flight control system, it was a default entry into a state (of the machine) which was inconsistent with the state of the environment. For both examples, the underlying approach and methods used to highlight these moding problems are briefly discussed.

INTRODUCTION

By a mode I understand the affection [states] of a substance, or that which is in another through which it is also conceived. (Benedict de Spinoza: *The Ethics*, 1677)

Human-machine interaction in supervisory control systems is a topic of much concern to operators, regulatory agencies, operational management, and manufacturers of such systems. Summary statistics show that crew-caused factors account for the majority of failures in supervisory control systems, and that this is consistent across a variety of domains: aviation, nuclear, and maritime (Lautman and Gallimore, 1988; Marine Board, 1994; Trager, 1988; Woods, Johannesen, Cook, and Sarter, 1994). In commercial aviation, human-machine interaction has received much attention lately because of several highly publicized accidents and incidents involving "glass-cockpit" aircraft (*Aviation Week and Space Technology*, 1995). Evidence indicates that the majority of factors contributing to these accidents and incidents occurred at the interaction between the crew and the automated flight control system—in particular, its various modes. This evidence has led to interest in modeling human-automation interaction, by certification and regulatory agencies, as well as some manufacturers of commercial aircraft (*Aviation Week and Space Technology*, 1995, Part 1, p. 64-65). The expected benefit of such modeling effort is to identify, early in the design phase, the potential for human-automation breakdowns in a given supervisory control system (Green and Richmond, 1993).

In the last two decades various models of human-computer and human-machine interaction have been suggested. Models such as GOMS (Card, Moran, and Newell, 1983; Kieras, 1988), Abstraction Hierarchy (Bisantz and Vicente, 1994; Rasmussen, 1986), and OFM (Jones, Chu, and Mitchell, 1995; Mitchell, 1987), focus on the human element in the context of a machine and/or environment. However, modern control systems operate in real time and are event driven—reacting to both external and internal stimuli. These reactions, which are part of any automated control system, place much cognitive demand on part of the human supervisor (Sarter and Woods, 1995). Nevertheless, most of the above models do not explicitly describe these reactions (Degani, Mitchell, and Chappell, 1995).

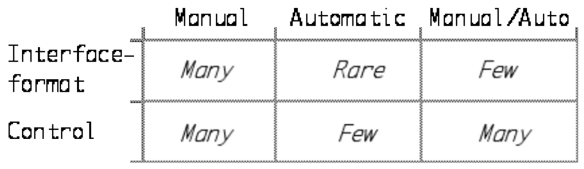
This paper is an introduction to a modeling framework called "OFAN"; it describes our initial efforts to model reactive systems in the context of both the operating environment and the functions/tasks of the human supervisor. The objectives of this paper are: (1) to suggest a language and a framework, (2) to use it to represent two examples of mode problems, and (3) to identify, via the model's formalism, the particular features of the human-machine-environment system that contributed to such breakdowns. One goal of our ongoing research effort is to classify and incorporate several of these breakdown-inducing features into a theory of mode problems. The purpose of the model is to serve as the language to describe the theory and as an analysis tool to identify these breakdown-inducing features in a given supervisory control systems.

MODES

Modes are implemented, in one way or another, in almost any modern machine or piece of equipment (Monk, 1986). Every control system that can perform a variety of functions has modes; the more functions, the more modes (Norman, 1981; 1983). Somewhat narrower than Spinoza's definition, we define here a mode as "the manner of behavior" of a given system. A machine may have various (and mutually exclusive) ways of behaving. Each behavior is defined by the machine's input, output, and states, as a function of time. And for each behavior, or mode, various types of inputs (e.g., constraints or target-values) may be imposed either by the operator or by the machine.

Based on our survey of mode implementations in supervisory control systems, we classify modes according to two attributes: their functions, and the types of transitions they allow. In terms of function, two types are described here: (1) for presenting different *Interface formats*, and (2) for allowing different *Control behaviors*. With respect to transitions, or the change from one mode behavior to another, we identified three types: (1) those that are commanded (engaged) by the operator—*Manual mode* transitions, (2) those commanded by the reactive system—*Automatic mode* transitions, and (3) both—either Manual or Automatic (*Manual/Automatic*).

These two attributes (function, transitions) are combined in Figure 1 to form a two by three matrix. Later, we use our framework to identify some of the unique features, typical to Interface format and Control modes, that actually induce mode problems.



	Manual	Automatic	Manual/Auto
Interface-format	*Many*	*Rare*	*Few*
Control	*Many*	*Few*	*Many*

Figure 1. Mode classification and observed frequencies (in our survey of modes).

PROPOSED FRAMEWORK

The framework suggested here, called OFAN, is based on our field study of operators (e.g., pilots) using modes. We hypothesize that there is a unique link between the structure of the environment, the operators' functions/tasks, and the mode structure of a given modal system. We use the term "mode structure" to describe the mode behaviors and mode transitions in a given machine. In a previous paper, describing how pilots transition between modes in actual flight operation (Degani, Shafto, and Kirlik, 1995), we identified a direct link between the states and events in the environment, and when and how pilots transition between modes. We argue that mode transitions are the emergent behavior of the interaction between (1) the environmental structure, (2) the pilots' procedures and techniques[1] , and (3) the mode structure of the system. Our framework,

[1] The contribution of procedures is beyond the scope of this introductory paper. (See Degani and Wiener, 1994, for a more complete treatment of this factor)

therefore, tries to represent this environment-pilot-machine relationship in order to identify mode problems.

Our thesis is that the structure of the environment should correspond to the mode structure of the machine, and that this relationship should be portrayed in the perceptual structure of the interface. When this is achieved, there is an opportunity for fluent interaction between the environment, the operator, and the machine. On the other hand, mismatches between the environment structure, the mode structure, and the interface perceptual structure demand cognitive resources and are therefore more susceptible to errors. (See Kirlik's 1995 ecological task analysis for a complete description of this thesis).

Models and Formalism

The OFAN framework suggested here is based on two distinct, yet related, modeling approaches: the operator function model (OFM) and Statecharts. Both share the same underlying theory—the finite state machine—which provides a medium for describing the control behavior of a given system in terms of its states and transitions (Minsky, 1967).

The OFM is a task decomposition formalism used for task analysis and simulation (Mitchell, 1987). Depending upon demands from the environment (e.g., ATC clearance) and the requirements of the task (e.g., mandatory procedures), the model specifies the progression of tasks and actions (either manual, perceptual, or cognitive) that a well-trained operator will execute. The OFM extends the traditional finite-state-machine theory by providing for both hierarchy and concurrency: Hierarchy is implemented by having several levels of description—represented by higher level nodes (functions) encapsulating sub-nodes (sub-functions, tasks, actions), while concurrency is implemented by allowing several nodes to be active simultaneously.

Statecharts is a specification language for complex, real-time, reactive systems (Harel and Pnueli, 1985). Statecharts formalism also allows for hierarchy and concurrency. Similar to the OFM, hierarchy is represented with the notion of a super-state and sub-states. Concurrency is provided by the notion of independence and orthogonality such that a super-state containing two orthogonal sub-states can be described as their product (.AND. relationship). Another feature of Statecharts is its broadcasting mechanism—broadcasting the active states and events (that trigger transitions) to the entire network. The Statecharts formalism allows for several other enhancements such as default entries into a state, conditional entries, selection- and history-based entries, and overlapping states (Harel, 1987).

These features of the OFM and Statecharts are well suited for describing human interaction with a modal system: (1) a finite-state-machine approach maps quite well for describing modes (Jacob, 1986), (2) hierarchy allows us to deal with the organization of modes and sub-modes within a system, (3) concurrency allows us to describe a multi-components system in which several modes are simultaneously active, and (4) broadcasting allows us to specify transitions and their effects on the system. Both the OFM and Statecharts represent the control behavior of the system—what is responsible for making the time-dependent decisions that influence the system's entire behavior (Harel, 1988).

OFAN Representation

The similarities between the OFM and Statecharts with respect to the underlying finite-state-machine theory and extensions (hierarchy and concurrency) provided a common ground for amalgamation of the two models. The OFAN framework uses the Statecharts language to describe the human-machine-environment system as five concurrently active modules: (1) the *Environment* (2) the *Human Functions and Tasks*, (3) the *Controls*, (4) the *Plant* [2], and (5) the *Displays*.

The *Environment* module describes the events in the operational environment as a collection of concurrent states. Although we realize that this view of the environment is somewhat naive, it serves well for feeding triggering events into the Human Functions and Task*s* module and the system (cf. Bråthen, Nordø, and Veum, 1994, for a similar approach). The *Human Functions/Tasks* module describes how the operator changes his/her activities (modeled as states) and the events that trigger them. This module is essentially an OFM in Statecharts clothing. The *Controls* module describes the system's control panel—its various states and transitions. The *Plant* module describes the control system—its components, states, inputs, and transitions. Closing the loop is the *Displays* module, describing the possible states and transitions in the accompanying display (or any other form of feedback to the operator).

The OFAN framework represents the complete environment-human-controls/displays-plant ensemble as a set of concurrently active modules—all interconnected, via triggering events, to form a whole. We hypothesize that a system which allows for fluent human-machine interactions, states and transitions should be relatively consistent among its modules—both in terms of structure (hierarchy, concurrency) and in terms of triggering events. This implies, of course, that any significant departure from this consistency is an indication of potential moding problems (cf. Kieras and Polson, 1985, for a similar approach in human-computer-interaction).

METHOD

Using the OFAN framework we describe here two systems that foster moding problems. These examples were collected in our field study of mode usage in glass cockpit aircraft. The first example describes a problem in an *Interface format* mode, the second describes a problem in a *Control* mode. For this analysis, three types of relevant information were collected: (1) data about how pilots transition among modes (2) problems they had with these modal systems, and (3) the software code.

EXAMPLE I: INTERFACE FORMAT MODE

This example was documented during cockpit observations in a highly automated glass-cockpit aircraft. The pilots complained that there was "something weird" in changing the

[2] From here on we use the term plant, instead of machine, to describe the control system. We reserve the term system to describe the complete human-plant-environment system.

range and format of the horizontal situational display. During an approach to the airport, this display is monitored by the pilots in order to assess the relationships between the aircraft (ownship) and the location of the objects within the environment—e.g., airport, waypoints, other aircraft in vicinity, thunderstorm cells, and more. Also, per procedures, the crew should monitor this display to assess the relationship between ownship and various aids such as ILS (instrument landing system), VOR (very high frequency omnidirectional radio), and/or ADF (automatic direction finder). After an examination of the controls and the display, it was evident that the source of the problem arose from an unexpected interaction between the mode and range-selector knobs.

Description

The range-selector knob allows the pilot to select among six range options (320 nautical miles, 160 nm, 80 nm, 40 nm, 20 nm, and 10 nm). The mode-selector knob (actually a format-selector) allows the pilot to switch among five different display formats: “PLAN,” “ARC,” Navigation (“NAV”), “VOR,” and “ILS” (see Figure 2). The PLAN format shows the route of flight in a North-up orientation; it is mainly used for reviewing the flight plan. The ARC format presents a 90-degree arc view (track-up orientation) emanating from the ownship symbol at the bottom of the screen; it is the most

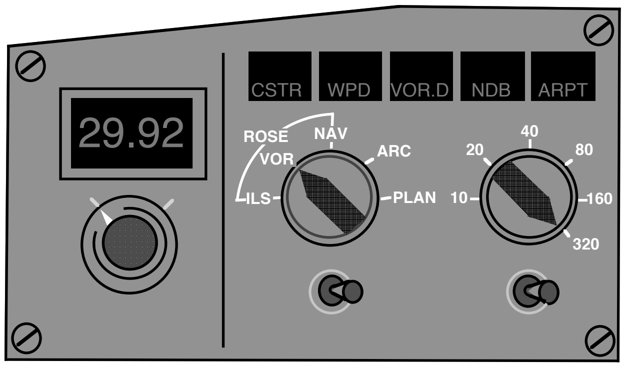


Figure 2. Mode- and range-selector knob.

frequently used display format. The NAVigation, VOR, and ILS formats all present a 360-degree view (rose) around the ownship symbol (located at the center of the display); these formats are normally used during an approach.

When in ARC format, the range selected (with the range-selector knob) corresponds to the distance between the ownship symbol to the edge of the arc. When in any of the “Rose” formats (NAVigation, VOR, and ILS), the range selected is not from ownship symbol to the edge—it is from the upper edge to the lower edge (see Figure 3). Therefore, when the pilot switches from ARC to any of the “Rose” formats, objects at the top of the display disappear (those that are more than half-way distant from ownship). For example, say the active format is ARC, the range selected is 20 nm, the destination airport is 15 miles, and then the pilot switches to NAVigation format—as a result, the airport symbol disappears! Manipulation of either the range selector (switch to 40 nm) or display format (switch back to ARC format) will bring the airport back to view.

Two additional factors augment this moding problem: One, the transition between ARC and any of the “Rose” formats usually occurs during a busy period in the flight—the approach to the airport. Two, after switching the range or format, it takes 1-3 seconds for the display to update itself (in the meantime, it is blank).[3]

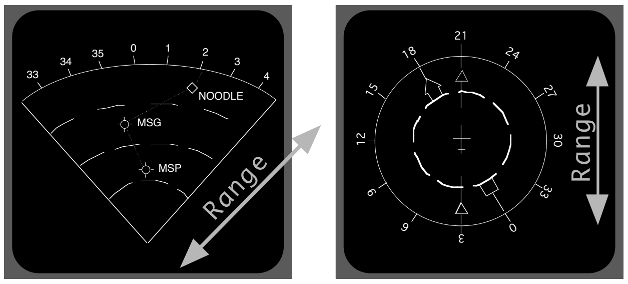


Figure 3. Arc- and Rose-mode formats.

Model

Figure 4 depicts the OFAN representation of this Interface format mode.[4] The *Environment* module (top) describes the range between ownship and airport as well as the reception of navigation aids by the aircraft sensors. Both are concurrent states of the environment (this relationship is depicted by using a broken line to separate the states). The *Human-Functions/Tasks* module describes two functions performed by the crew—(1) review of planned route and (2) monitoring of the relationship between the aircraft and objects in the environment. Both are exclusive states, or activities, of the operator (this relationship is depicted by using rounded rectangles to represent states). While in the monitoring function, the crew must update the range, and monitor the navigation aids as well.

The *Controls* module presents the range and format knobs as concurrent super-states, each one with its own set of sub-states (6 ranges and 5 formats). Transitions from one state to another are initiated by rotating the knob; the triggering events for these transitions come from the *Human-Functions/Tasks* module (*Update Range* activity). The *Plant* module also has two concurrent super-states, range and format, each with its own sub-states. Transitions between sub-states are triggered by events (rotating the knob) in the *Controls* module. For example, switching the range-selector knob from 20 nm to 10 nm (depicted as transition c10) triggers an event in the *Plant* module (p10)—this relationship is depicted as c10/p10. Closing the information flow loop is the *Displays* module. Again the same structure is repeated: two concurrent super-states, and transitions that are triggered by transitions in the *Plant* module. For example, event p10 in the *Plant* module triggers a transition (d10) that switches the display range from 20 nm to 10 nm—this relationship is depicted as p10/d10. Another element in specifying transitions is a parenthesized condition or a guard, e.g., d11(transition to "Rose" mode)—indicating that transition d11 will occur only when the transition to Rose mode state has fired.

[3] There is, however, one advantage to this design—the display resolution stays constant regardless of format changes.

[4] The formal representation in Figure 4 is for illustration purposes only—it is in no way meant to be complete.

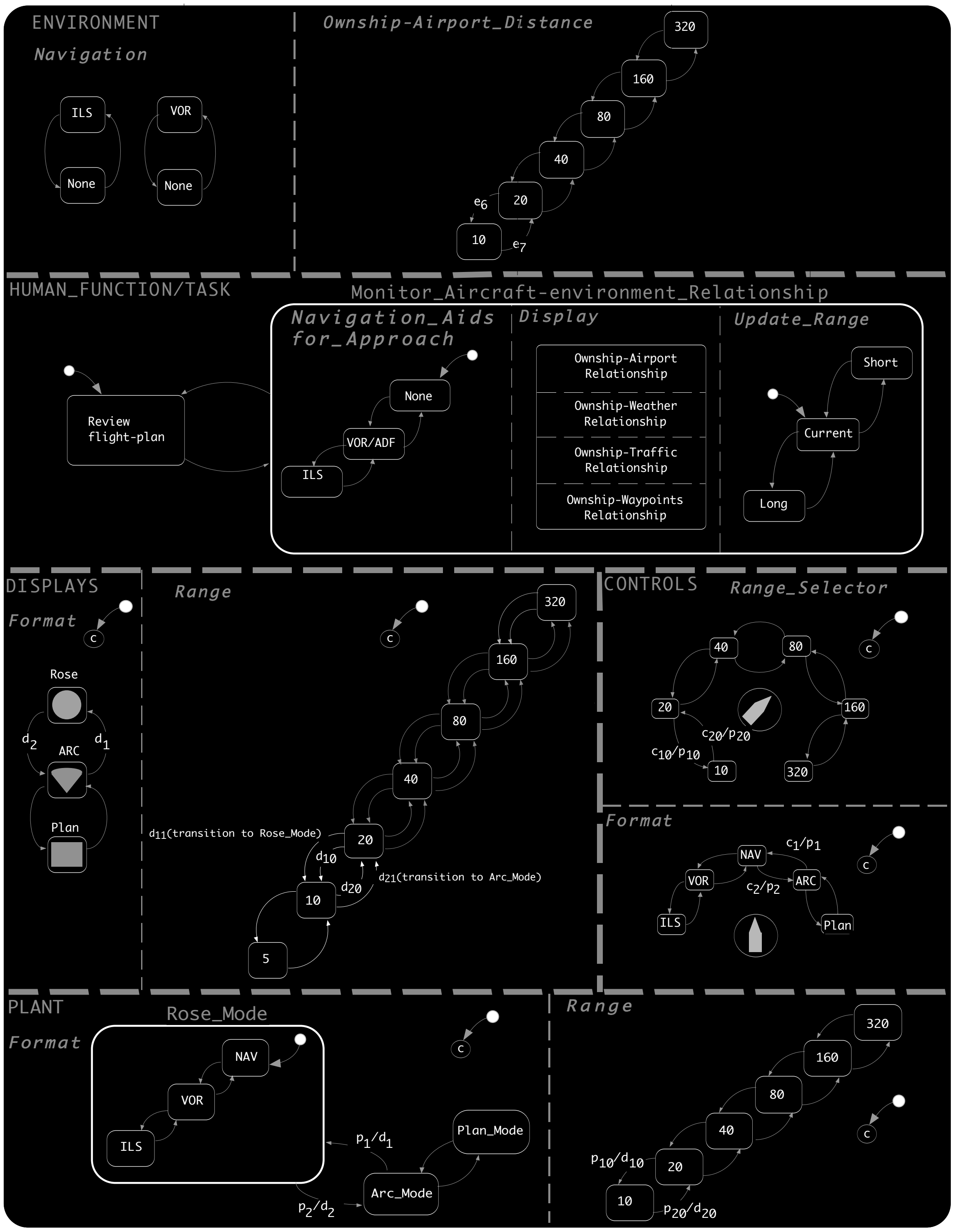


Figure 4. *An Interface format mode example.*

Analysis

The model shows the progression of events in the environment, human, controls, plant, and displays. Events within a given module are mostly dependent upon events occurring in other modules. In addition, the structure of super-states and transitions is maintained throughout the five-module system: two independent super-states and a set of sub-states. However, a closer look at the *Display*s module indicates some departures from this consistency. First, there are two arcs for transitioning from one range to another: one arc corresponds to transition in the *Controls* and *Plant* modules—i.e., rotating the knob; the other is not—it is predicated upon transitioning to “Rose” format. This dependency, however, is not intuitively grasped from the control-panel layout, and it is not consistent with the independence between the range and format super-states that exists in the *Environment*, *Human Functions/Tasks*, and *Plant* modules. Another interesting aspect of this dependency is the “birth” of a new range: 5 nm. This range, not specified on the range-selector knob, arises from the interaction between the 10-nm range and “Rose” format. This “new” range appears, however, only when we model the display from the point of view of the pilot’s task.

The pilot’s task (as modeled in the *Human Functions/Tasks* module) is to monitor the relation between ownship and the objects in the environment. During the transition between ARC and any of the “Rose” modes, the display provides the pilot with a *consistent* relationship between objects in the environment (say waypoint A and waypoint B), and an *inconsistent* relationship between *ownship* and waypoints A and B. This semantic difference, with respect to the pilots’ intended functions, fuels this particular moding problem. We hypothesize therefore, that one feature of mode problems is due to such reachability issue—two transitions to the same state. One transition is consistent with the layout of the controls, the human functions, and the environment, while the other is not.

EXAMPLE II: CONTROL MODE

This example was documented during one cockpit observation in another glass-cockpit aircraft. The aircraft was climbing to 11,000 feet per air traffic control (ATC) instructions, and a fully automatic vertical mode called “Vertical Navigation” was active. In this mode the speed target value is obtained from the flight-management computer, which computes the most economical speed given the situation (about 300 knots in this case). During the climb (at about 10,500 feet), ATC instructed the crew to reduce speed to 240 knots. The first officer engaged a unique feature of Vertical Navigation called “speed intervene,” which allowed him to enter the speed, specified by ATC, via the mode-control panel as a new target value (see Figure 5).

As the aircraft neared 11,000 feet, it started the level-off maneuver by automatically disengaging Vertical Navigation and engaging “Altitude Capture” mode. Once at 11,000 feet, the Altitude Capture mode disengaged automatically and the “Altitude Hold” mode was engaged automatically. During and after the maneuver, the speed was kept at 240 knots. Shortly after, ATC instructed the crew to climb to 14,000. The first officer reached up to the mode-control panel and engaged the Vertical Navigation mode in order to initiate the climb. However, instead of climbing at a speed of 240 knots (as was still required by ATC) the aircraft speed defaulted to 300 knots! The crew violated the ATC speed restriction because they assumed that Vertical Navigation mode would “remember” the target value entered previously into the mode-control panel. However, the Vertical Navigation mode, once re-engaged, defaults to the economy speed computed by the flight-management computer.

Description

The automatic flight control system of this aircraft has several modes to control the vertical aspect of the flight (see Figure 5). Three modes are discussed here: Altitude Capture, Altitude Hold (ALT HOLD), and Vertical Navigation (VNAV). With respect to speed target-values (an input that affects mode behavior), the Altitude Capture and Altitude Hold modes obtain this input from the mode-control panel. By default, the Vertical Navigation speed target value is obtained from the flight-management computer (which computes economy speed). Yet another option, called "speed intervene," allows the pilot to override the flight-management computer speed and enter a different speed. This is achieved by pressing on the speed knob (located on the mode-control panel) and then dialing in the desired speed.

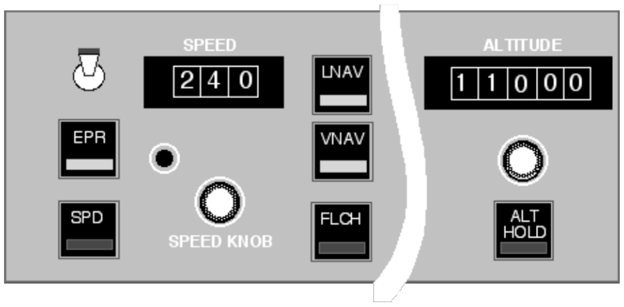


Figure 5. Mode-control-panel.

Model

Figure 6 depicts the OFAN representation of this example.[5] The *Environment* module describes the demands (in this case, ATC instructions) as a vector of several elements. Within each element, we describe its possible attributes as sub-states. For example, the speed can be either unrestricted or restricted. Speed restrictions can be either instructed directly by the ATC controller or mandated by a given procedure (e.g., a standard arrival procedure). The *Human Functions/Tasks* module describes two functions performed by the crew: (1) modify, and (2) monitor. The modify function describes the various activities (e.g., modify speed, change altitude, etc.) as sub-states. The monitor function describes the various orthogonal monitoring activities as states. For example, the monitor speed activity has two sub-states: (1) deceleration/acceleration—specifying the monitoring tasks when the aircraft speed is changing, and (2) constant—specifying the monitoring tasks when the aircraft speed has reached a steady state.

The *Controls* module depicts the mode-control panel and its various controls: (1) the speed knob which, by pushing it, engages the "speed intervene" feature, and (2) buttons for manually engaging modes. The Plant module represents the three modes discussed in this example: Altitude Capture, Altitude Hold, and Vertical Navigation. Each mode is represented by two orthogonal sub-states: one describing the control behavior of the mode (the control algorithms used), and the other describing two sources of inputs, or constraints—speed and altitude.

[5] Again, the formal representation in Figure 6 was not intended to be complete.

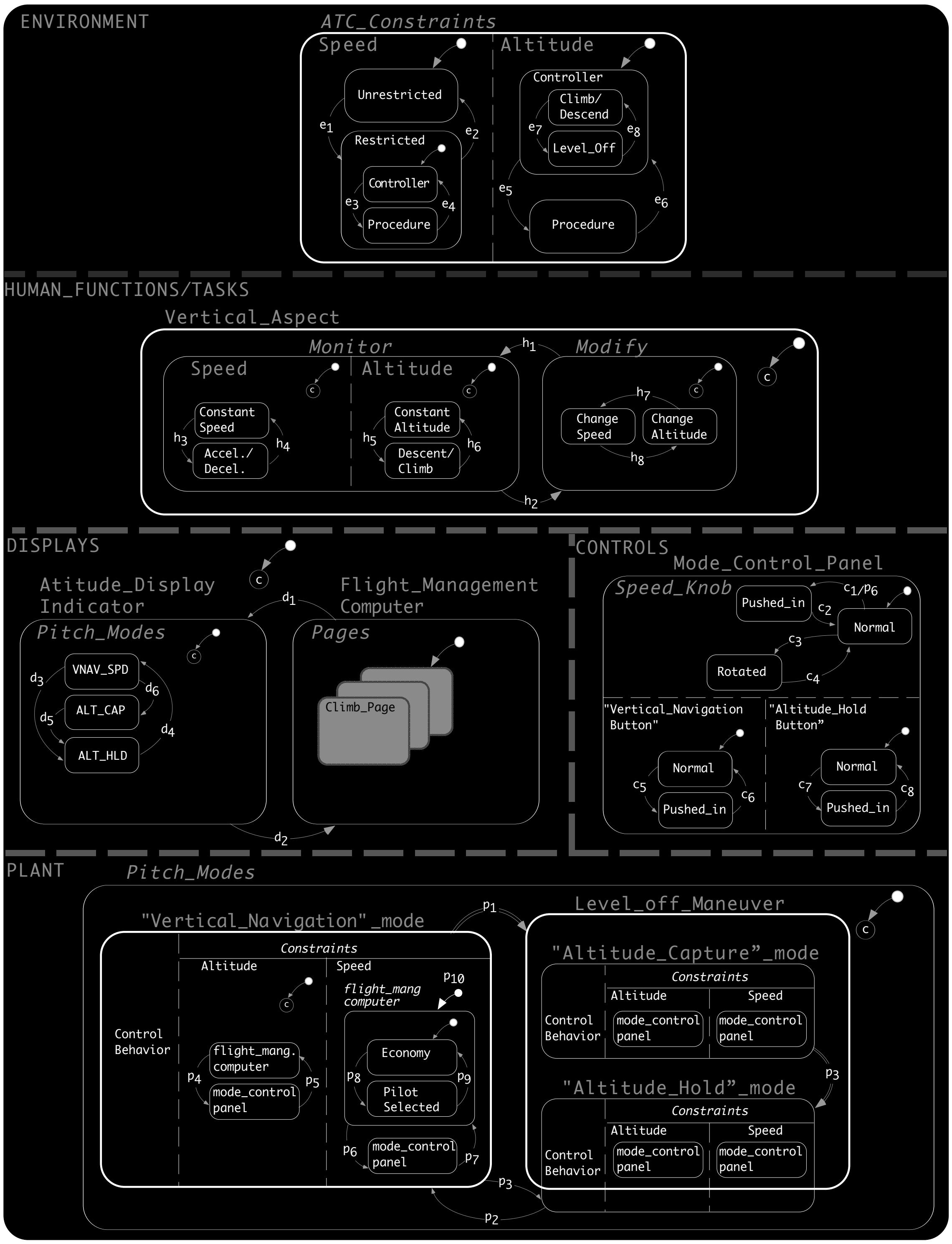


Figure 6. *A Control mode example.*

For example, when in Vertical Navigation mode, the speed target value is either obtained from the flight-management computer (default entry—depicted as a small arrow) or from the mode-control panel. When the source of target-value is the flight-management computer, the speed can be either "computed" (economy speed) by the computer or "selected" (entered manually into the computer) by the pilot. Transitions, either Automatic or Manual, in the *Plant* module are between (1) Vertical Navigation mode and the level-off state—an Automatic transition (depicted as two parallel lines), (2) Altitude Capture mode and the Altitude Hold mode—also an Automatic transition, and (3) Altitude Hold and Vertical Navigation mode—a Manual transition. The Altitude Hold mode exists both within the level-off maneuver and as an independent mode (overlapping state), indicating that this mode can be engaged in two ways: (1) automatically by the level-off maneuver, and (2) manually by the pilot. Lastly, the *Displays* module describes the annunciation of the mode status (on the attitude-indicator-display) and the climb page of the flight-management computer which displays the speed target value.

Analysis

The mode problem occurred when ATC instructed the crew to climb from 11,000 to 14,000 feet. At that point, the crew *re*-engaged the Vertical Navigation mode. However, they forgot to *re*-enter the target-value speed (240 knots). As a result, a speed violation occurred.

Once the automated flight control system transitions from any mode to Vertical Navigation mode, the speed target value defaults to the economy speed (in this example, 300 knots). This is represented in the Vertical Navigation mode description (*Plant* module), as the default entry into flight management system (p10). The result of this discrepancy—between the expected behavior of the mode versus the observed behavior—was a speed violation on part of the crew.[6].

The problem is detected when we note a mismatch between the *Plant* state and the *Environment* state. While the environment did not change any of its states (i.e., speed), the *Plant* did. In other words, the necessary correspondence between the state changes in the *Environment*, *Human Functions/Tasks*, and *Plant* modules is violated. This moding problem, we argue, occurs because of a non-fluent mode transition (Palmer, 1995)—the transition causes a state change that deletes a constraint which was previously entered by the pilot. In addition, the structure of the controls/displays do not allude to this default entry into economy speed. In ecological task-analysis terms (Kirlik, 1995), there is a mis-specification between the perceptual surface structure of the controls/displays and the opportunities for action (mode engagements and entry of target values). To overcome this mismatch, pilots must resort to cognitive resources (e.g., long-term memory). These resources, as can be seen from this incident, are susceptible to retrieval limitations.

CONCLUSIONS

In this paper, we suggested and described a framework for representing human interaction with modal systems. The underlying concept behind the structure of the framework and its five modules is the concept of independence and dependence. Independence, as the five modules (i.e., *Environment*, *Human Functions/Tasks*, *Controls*, *Plant*, and *Displays*) are all active, separately, at the same time. Dependence, because

[6] We use the term discrepancy, in lieu of the term error, because the latter, although commonly used in statistics and control theory, implies the assignment of blame when discussed in the context of humans.

transitions within a given module are contingent upon the events in another. At every "step" (initiated with the arrival of an event), the entire model resonates until no more internal events and transitions are fired—at this point the system settles down to a steady state.

Using the OFAN framework we were able to represent the mode structure, human functions/tasks, and the operating environment of two systems. Such representation of human-automation interaction is portrayed in a language which is commonly used by control engineers (Leveson, Heimdahl, Hildreth, and Reese, 1994). Furthermore, when the plant's behavior is already specified in a finite-state-machine formalism (such as Statecharts), the costly overhead, associated with building a human-machine model from scratch, is much reduced.

Armed with these two representations of modal systems, we were able to identify two intrinsic features that induce mode problems: The first—a problem associated with an *Interface format* mode—was the product of dual transitions into a state: one consistent with the controls layout and the environment structure (and therefore intuitive), the other dependent upon some internal state (and therefore unintuitive). The second—a problem associated with a *Control* mode—was a default entry into a state (of the machine) which was inconsistent with the state of the environment. It represents a non-fluent mode transition with respect to the speed input.

By incorporating these as well as other features that induce mode problems into a theory or a "catalog," one may develop a template. Our hope is that such a template can be employed to identify potential mode problems when it is most needed—during the initial design phase of a new human-automation system.

ACKNOWLEDGMENTS

This work was funded by NASA's Aviation Safety and Automation Program. The first author was supported by Cooperative Agreement NCC 2-798 between the San Jose State University Foundation and NASA Ames Research Center. Part of this research was conducted at the Center for Human Machine System Research (CHMSR) at Georgia Institute of Technology, Atlanta. We gratefully acknowledge Christine M. Mitchell for her support and encouragement during the development of this framework. James A. Lockhart provided valuable help with the graphical aspects of this framework. We thank Peter G. Polson, Michael G. Shafto, Stephen M. Casner, and Everett A. Palmer, for their comments on early drafts of this paper.

REFERENCES

Aviation Week and Space Technology. (1995). Automated cockpits: who's in charge? Part 1 and 2. 142(5 and 6).

Bisantz, A. M., and Vicente, K. J. (1994). Making the abstraction hierarchy concrete. *International Journal of Human-Computer Studies, 40*, 83-117.

Bråthen, K., Nordø, E., and Veum, K. (1994). An integrated framework for task analysis and system engineering: approach, example, and experience. *International Journal of Human-Computer Studies, 41*, 509-526.

Card, S. K., Moran, T. P., and Newell, A. (1983). *The psychology of human computer interaction*. Hillsdale, NJ: Lawrence Erlbaum.

Degani, A., Mitchell, C. M., and Chappell, A. R. (1995). Task models to guide analysis: Use of the operator function model to represent mode transitions. In R. S. Jensen (Ed.), *Proceedings of the Eighth International Aviation Psychology Symposium Conference*. Columbus, OH: The Ohio State University.

Degani, A., Shafto, M., and Kirlik, A. (1995). Mode usage in automated cockpits: Some initial observations. In T. B. Sheridan (Ed.), *Proceedings of the International Federation of Automatic Control; Man-Machine Systems (IFAC-MMS) Conference*. Boston, MA: IFAC.

Degani, A., and Wiener, E. L. (1994). *On the design of flight-deck procedures* (NASA Contractor Report 177642). Moffett Field, CA: NASA Ames Research Center.

Green, B., and Richmond, J. (1993). Human factors in workload certification. *Proceedings of the Aerotech '93 Conference*. Society of Automotive Engineers.

Harel, D. (1987). Statecharts: A visual formalism for complex systems. *Science of Computer Programming, 8*, 231-274.

Harel, D. (1988). On visual formalisms. *Communications of the ACM, 31(5)*, 514-530.

Harel, D., and Pnueli, A. (1985). On the development of reactive systems. In K. R. Apt (Ed.), *Logics and models of concurrent systems* (pp. 477-498). Berlin: Springer-Verlag.

Jones, P. M., Chu, R. W., and Mitchell, C. M. (1995). A methodology for human-machine systems research: Knowledge engineering, modeling, and simulation. *IEEE Transactions on Systems Man and Cybernetics, SMC-25(7)*, 1025-1038.

Kieras, D. E. (1988). Toward a practical GOMS model methodology for user interface design. In M. Helander (Ed.), *Handbook of human-computer interaction* (pp. 135-157). Amsterdam: Elsevier Science.

Kieras, D. E., and Polson, P. G. (1985). An approach to the formal analysis of user complexity. *International Journal of Man-Machine Studies, 22*, 365-394.

Kirlik, A. (1995). Requirements for psychological models to support design: Toward ecological task analysis. In J. M. Flach, P. A. Hancock, J. K. Caird and K. J. Vicente (Ed.), *An ecological approach to human machine systems I: A global perspective*. Hillsdale, NJ: Lawrence Erlbaum.

Lautman, L. G., and Gallimore, P. L. (1988). *Control of the crew-caused accidents*. Seattle: Boeing Commercial Airplane Company.

Leveson, N. G., Heimdahl, M. P. E., Hildreth, H., and Reese, J. D. (1994). Requirements specification for process-control systems. *IEEE Transactions on Software Engineering, 20(9)*, 684-707.

Marine Board. (1994). Risk, the operating environment, and safety. In National Research Council (Ed.), *Minding the helm* (pp. 159-186). Washington DC: National Academy Press.

Minsky, M. L. (1967). *Computation: Finite and infinite machines*. Englewood Cliffs, NJ: Prentice-Hall.

Mitchell, C. M. (1987). GT-MSOCC: A domain for research on human-computer interaction and decision aiding in supervisory control systems. *IEEE Transactions on Systems, Man and Cybernetics, SMC-17(4)*, 553-572.

Monk, A. (1986). Mode errors: a user-centered analysis and some preventative measures using keying-contingent sound. *International Journal of Man-Machine Studies, 24*, 313-327.

Norman, D. A. (1981). Categorization of action slips. *Psychological Review, 1(88)*, 1-15.

Norman, D. A. (1983). Design rules based on analysis of human error. *Communication of the ACM, 28(4)*, 254-258.

Palmer, E. A. (1995). “Oops, it didn’t arm”—A case study of two automation surprises. In R. S. Jensen (Ed.), *Proceedings of the Eighth International Aviation Psychology Symposium Conference*. Columbus, OH: The Ohio State University.

Rasmussen, J. (1986). *Information processing and human machine interaction: An approach to cognitive engineering*. New York: Elsevier .

Sarter, N. B., and Woods, D. D. (1995). How in the world did we ever get into that mode? Mode error and awareness in supervisory control. *Human Factors, 37(1)*, 5-20.

Trager, E. A. (1988). *Special study report: Significant events involving procedures* (Office for Analysis and Evaluation of Operational Data AOED/S801). Washington DC: Nuclear Regulatory Commission.

Woods, D. D., Johannesen, L. J., Cook, R. I., and Sarter, N. B. (1994). *Behind human error: Cognitive systems, computers and hindsight*. Dayton, OH: Crew Systems Ergonomic Information and Analysis Center.